GRAPHICS DRAWING COMPILER-PET AND SYM

1. GENERAL

This Graphics Drawing Compiler is composed of a number of macros developed to be used with C. Moser's Macro ASSM/TED to convert the assembler into a compiler. The main purpose of this work, is to illustrate by example the anatomy of an easy to understand compiler, and to provide a mechanism whereby the reader could easily develope his own compiler be it an industrial control compiler, music compiler, or just a collection of macros which aid program development. Although these macros do not provide an extensive graphics drawing language, they do lay the ground work for those of you who would like to add to this language or rewrite it.

When the graphics drawing macros have been entered into the ASSM/TED's text file, the Macro ASSM/TED is converted into a Graphics drawing compiler, Programs can still be written in 6502 assembly only, in the graphics drawing language only, or a combination of both.

For those who are not familiar with the term, a compiler is a program which translates statements written in a high level language into a sequence of machine instructions. Since this compiler generates pure machine code, no runtime package is required. In fact, after you have sucessfully compiled a program, it can be executed without the ASSM/TED and the graphics drawing macros.

Those who are really into graphics will find their programs will draw images many times faster than an equivalent program written in BASIC. If desirable it is possible to write part of your program in the graphics drawing language and the rest in Basic. Several extensions to the Macro ASSM/TED are provided in this document to make it easy for the user to use the compiler. They are:

1) ≥BUILD command to build a compiler or label library
2) Provision for the ≥FORMAT command to set the maximum number of characters per label. This is useful especially since the PET has only 40 characters per line display'
3) A patch to make ASSM/TED and PET BASIC coexist without destroying each others zero-page variables.

A cassette was shipped to you which contained the Graphics Drawing Macros and an example program which draws a 3-dimensional box on the screen.

Remember, whether you intend to use this information for graphics drawing or for some other macros implementation, the ideas presented apply to practically all applications. Macros can represent high-level interface between the programmer and assembly language, making the source listing easier to read. Thus coding should be easier for the programmer resulting in programs which are more reliable and less expensive to produce.

2. GRAPHICS COMPILER INSTRUCTION SET

A description of each instruction in the graphics drawing language is presented in this part. All argument parameters are either addresses or data. If the arguments are data, or addresses which point to data, the operations performed will be on single byte quantities. For example, the ADD and SUB instructions perform arithmetic on single byte quantities.

Most of the arguments in these instructions are symbolic or non-symbolic address quantities. Two instructions, SET and DEFINE, allow one to store a quantity at a specified location. If the quantity is non-symbolic, then that quantity is stored at the specified location. But, if the quantity is symbolic then the lo part of the address is stored at the specified location.

Therefore, the following is used to distinquish between address and quantity:

label1 label2...etc = symbolic or non-symbolic address,

#1 #2...etc = data quantities.

The Graphics Drawing Compiler instruction set follows:

ADD (label1 label2)

Add the contents at label2 to the contents at label1 and store the result at label1. This is a one byte addition operation.

BEGIN

Begin Graphics Drawing Compilation. Each graphics drawing program must have exactly one of the statements and it must be the first executable instruction.

BELL

Ring bell or some user provided audible device. The user may provide software driver and hardware to accomplish this. See BELL subroutine in BEGIN statement.

The BELL instruction for the SYM, causes the on board audible device to beep.

For the PET, the BELL instruction enables the serial I/O shift register and provides a signal on the CB2 lead (pin M) of the parallel user port.

CLEAR

Clear screen from current cursor position to end.

DO (labelI label2)

Set up do loop to loop until next END instruction. The number of times the loop is to be performed is contained at location label2. On completion of the DO loop, go to labelI.
Example:

```
          DEFINE (J 10)   ;Set J=10
          DO  (EXIT J)    ;loop 10 times
            .              ;then go to EXIT
            .
            .
          END
```

Common errors: Entering non-symbolic labels such as DO (EXIT 4), not terminating with END, making labelI point to within a DO loop.

DEFINE (labelI #1)

Store the one byte quantity #1 at location labelI.
Example:

```
          DEFINE (COUNT 4)
```

Common errors: Entering symbolic labels where non-symbolic is required and vice versa, not defining the label via .DE,.DI,or.DS

```
DRAWD ┐
DRAWL  >  (labelI  label2)
DRAWR  |
DRAWU ┘
```

DRAWD- Draw line down from current cursor position.
DRAWL- Draw line left from current cursor position.
DRAWR- Draw line right from current cursor position.
DRAWU- Draw line up from current cursor position.

Where:
labelI = location of character to use to draw the line.
label2 = location of the length of the line.

Example:

```
          DEFINE (CHAR 68)
          DEFINE (LENGTH 15)
          DRAWD  (CHAR LENGTH)
```

END

Terminates DO loops and/or program. Each DO loop must be terminated with its own END, and all programs to be executed via the >RUN command should be terminated with END or RTS.

Common Errors: Too many or too few End statements.

GRAPHN

Graphics Mode No. Exits screen graphics mode.

GRAPHY

Graphics Mode Yes. Enters screen graphics mode.

HOME

Home cursor (move to upper left corner of screen).

INPUTB (labelI) input from keyboard two hex digits and store at byte located at labelI.

INPUTC (labelI) input from keyboard one ascii character and store at labelI.

JUMP (labelI)

Jump to labelI.

```
JUMPE   ┐
JUMPG   │
JUMPGE  │
JUMPL   ├ (labelI  label2)
JUMPLE  │
JUMPN   ┘
```

Jump conditionally to label2 depending on quantity stored at labelI.

```
JUMPE - Jump if quantity at labelI =0
JUMPG - Jump if quantity at labelI >0
JUMPGE- Jump if quantity at labelI >=0
JUMPL - Jump if quantity at labelI <0
JUMPLE- Jump if quantity at labelI <=0
JUMPN - Jump if quantity at labelI ≠0
```

```
OUTPUTB  (labelI)  Output the byte at location labelI
OUTPUTC  (labelI)  Output the ascii character at location labelI.
```

POSABS (labelI label2)

Position cursor at absolute position on screen. Absolute coordinates are stored at labelI (row) and label2 (column).

If you specify labelI greater than 23 or label2 greater than 39, they will be respectively divided by 24 and 40 to obtain proper coordinates.

Note: 0 0 is home position and 23 39 is lower right corner.

Example: Position to column 18 of top row:

```
SETAB (0 18)
POSABS (↑A ↑B)
```

POSREL (labelI label2)

Position cursor relative to current position. Relative cordinates are stored at labelI (row) and label2 (column).

```
Examplel:  To position 4 rows down and 12 columns right from current
           position:
                         SETAB  (4 12)
                         POSREL (↑A ↑B)

Example2:  To position 1 row up and 6 columns left form current position:
                         SETAB  (24-1  40-6)
                         POSREL  (↑A  ↑B)
```

Note: To position up and left, you have to incorporate a wrap around count. The screen has 24 lines and 40 columns. If you position right 34 (40-6) then you move cursor to far right and back around for completion of count. This feature applies also for positioning relatively up.

PRINT (labelI)

Print the text at location labelI on the screen. The text may be set up using the .BY pseudo op, and should be terminated with a 00 byte.

```
Example:  to output the message "Input your next move?"
             PRINT  (MESSIN)
             .
             .
   MESSIN    .BY 'INPUT YOUR NEXT MOVE?' 0
```

Common Errors: Not terminating message with 00 byte, placing message text in machine instruction area of program.

REVRSN

Reverse Video No. Exits screen reverse video.

REVRSY

Reverse video mode yes. Enters screen reverse video.

```
SETA    (#1) store quantity #1 at location ↑A
SETAB   (#1 #2) store #1 at location ↑A, #2 at ↑B
SETABC  (#1 #2 #3) store #1 at location ↑A, #2 at ↑B, #3 at ↑C
SETABCD (#1 #2 #3 #4) store#1 at location ↑A, #2 at ↑B, #3 at ↑C, #4 at ↑D
```

Labels ↑A, ↑B, ↑C, and ↑D are predefined (via.DE) by the compilers BEGIN statement.

SUB (labelI label2)

Subtract contents at label2 from contents at labelI and store result at labelI. This is a one byte subtraction operation.

Common Errors: Entering non-symbolic labels

```
VECTUR ┐
VECTUL │(labelI label2 label3 label4)
VECTLR │
VECTLL ┘
```

VECTUR - Draw vector to upper right
VECTUL - Draw vector to upper left
VECTLR - Draw vector to lower right
VECTLL - Draw vector to lower left

Where:
labelI = location of character used to draw the vector
label2 = location of the "rise" quantity of the vector
label3 = location of the "run" quantity of the vector
label4 = location of the length of the vector

Example: Draw vector to upper right using character "A", 45 degree angle, and length of 10.

```
          SETABCD  ($41  1  1  10)    Note: rise to run of 1:1
          VECTUR   (↑A  ↑B ↑C  ↑D)          is 45 degrees.
```

3. ENHANCEMENTS TO ASSM/TED

As previously mentioned, this document provides three enhancements you can make to ASSM/TED. Two of these enhancements provide the following commands:

```
        ┌ MACROS
>BUILD <  LIBRARY     n
        └ CLEAR
```

>BUILD MACROS n Build into ASSM/TED a set of macros which can be used to define a compiler. This locks the macro definitions in the text file and its associated labels in the label file. n specifies the line number of the last line in the macro set which defines the compiler. You will note that if you type >PRINT after building a compiler, the macros will not be output.

>BUILD LIBRARY Build a library of labels in ASSM/TED's label file. This capability is not required for use with the Graphics Drawing Compiler but was provided as an additional feature for those who write programs which makes references to your microcomputers ROM entry points and special variables. Thus you can enter a program which has nothing but label definitions (with the last line a .EN), type > ASSEMBLE, then >BUILD LIBRARY, and you have locked these label definitions in the label file. Now you don't have to look up and define the labels for subsequent program assemblies.

>BUILD CLEAR Unbuild a previously entered ≥BU M or ≥BU L.

¦30 ERROR A ¦30 error message will be output if you try to build a set of macros or library when a build is already in effect. This error will also occur if you try to unbuild with no build in effect.

```
>FORMAT   ( SET   )  n
          ( CLEAR )
```

This is an enhancement to an existing ASSM/TED command. The >FORMAT SET n form allows the user to specify the maximum label length. For example, the default length is set by ASSM/TED at 10 characters/label. Many microcomputers have 40 character/line displays which do not leave very much room for the mnemonic and operand to appear on the same line. Thus, one could enter >FORMAT SET 4, get 4 characters per label and allow more space for the mnemonic and operand. The maximum allowable entry for n is 31.

The third enhancement is a provision for PET BASIC and ASSM/TED to coexist simultaneously. You may already know that PET BASIC "hogs" practically all of the zero page memory locations, leaving very few for other programs to use. Macro ASSM/TED needs 64 zero page locations for its own work, and currently both systems "tromp" on each others variables resulting in the PET hanging up if you exit ASSM/TED and go to BASIC.

This can be arbitrated by making ASSM/TED save BASIC's zero page variables when ASSM/TED is entered, and restoring these variables and saving its own when you exit ASSM/TED. Thus, a zero page swap area is maintained at 1E00 - 1EFF.

This zero page swap idea was courtesy of Bill Seiler - CBM.

To provide for these enhancements, enter the object code from the appropriate part of listing I (Ia for PET, Ib for version 1.0 non-PET, and Ic for version 2.0 non-PET). Note: You have version 2.0 if the message "C 1979 By C.MOSER" appears on cold start, else you have version 1.0.

After entering this object code, you may want to make a backup copy on tape or disc.

→ Note: After entering these enhancements, you should do a "cold start" entry in ASSM/TED so that various variables can be initialized.

4. OPERATIONS

A. Loading the Graphics Compiler Macro Set

First load the Macro ASSM/TED and begin execution. Allocate approximately 6K for the text file and 2K for the label file. Next, insert the supplied cassette in the tape deck and type >GET.

B. Build the Compiler

With the graphics macros loaded, type >AS and then >BUILD MACROS 4999. The number 4999 is the last line number in the macro set. If you omit 4999, you will lock into the text file the graphics drawing macros and everything after it. To examine what exactly is going on, type >SET and notice that the text file and label file starting addresses have changed. These now point to after the macro set locking the macros in the text and label files.

If you want to unbuild the macros and examine or make modifications, type >BUILD CLEAR. Again type >SET and note that the file boundaries are changed back to their original contents. If you did not alter the text file or performed any subsequent assemblies, you can rebuild the macros via >BUILD MACROS 4999.

→ If you altered the text file or performed an assembly, you will need to reassemble before rebuilding (>AS then >BU M 4999).

→ If you try to build a compiler already built or unbuild one that is not built, the !30 error will be output.

C. Creating a Graphics Program

The supplied cassette contains a program which draws a 3-dimensional box. To print this program on the screen, type >PRINT. If you want to enter some other program, type >CLEAR and enter your program. (If you type >CLEAR when a >BUILD MACROS is in effect, you clear only the text file following the macros.)

→ Note: Do not change the file boundaries (via >SET) if you have a build in effect.

D. Compiling

To compile a graphics program, insure that you have a .EN as the last line. Then type >ASSEMBLE. It will take a little longer to compile a graphics program versus a machine language program because many machine language instructions are being generated for each source line. To illustrate, compile and list (>AS LIST) and then observe the output.

E. Execution

The easiest way to execute a program is via the >RUN command. You should though insure that the last executable statement in your program is one of following: END, RTS, or JUMP to warm start in ASSM/TED.

For example, to run the 3-D Box program, type >RUN BOX. The message "INPUT HEIGHT THEN WIDTH" will appear. Respond with hex numbers for the height and width of the box to be drawn. Try 0A and 18 as an initial test and then experiment with other values. A listing of this program is shown in listing 2A for PET and 2B for SYM.

5. Useful Details of this Language

a) Each program must contain a BEGIN instruction as the first executable statement.

b) The compiler will define 4 variables (↑A,↑B,↑C,↑D,) which can be assigned values thru either of the following: SETA,SETAB,SETABC, SETABCD, or DEFINE. If you want to use some other variable, you will have to assign it storage via the .DE, .DI, or .DS pseudo ops and assign values via DEFINE. Note that ↑A,↑B,↑C,↑D, can be more convenient to use in that the SET graphics instruction class can assign values to more than one variable at a time.

c) Always terminate each DO loop with its own END instruction, and do not jump into the middle of DO loops.

d) If an error message other than !30 occurs, consult the ASSM/TED manual.

e) Avoid using labels in which the first character is an "↑" (example: ↑LOOP). The reason is the compiler macros generate a number of labels beginning with "↑" and if you define one of these in your program, a duplicate error message (!06) will occur.

6. ADDING YOUR OWN MACRO EXTENSIONS

You can add your own macros to this compiler by simply writing and entering then as described in the Macro ASSM/TED manual.

As an example, assume you want to write a game program which moves a car across the screen. You will need two macros: One to draw the car relative to the current cursor position, and another to clear the area around the current cursor position. Thus one could draw the car, clear it, move the cursor, draw it again, etc. to give the illusion of motion. The easiest way to define these macros is to incorporate an existing one-the PRINT statement. To draw the car, have the PRINT statement print it. To clear the car, have the PRINT statement output spaces. Thus the macros could be:

```
                                   enter code for cursor down
                                    8 is backspace or cursor left
;DRAW CAR
!!! CAR .MD
       PRINT (...CAR)                        enter code for cursor up
       JMP ...SKIP
...CAR .BY '▄█▄'  CD 8 8 8 'O O' 8 8 8 CU 0
...SKIP .ME          graphics characters which draws
;CLEAR CAR
!!!CLRCAR .MD
          PRINT (...CLR)
          JMP ...SKIP
...CLR    .BY '   ' CD 8 8 8 '   ' 8 8 8 CU 0
...SKIP   .ME
```

Now to draw the car and move it 2 positions, you could write:

```
          CAR
          CLRCAR
          POSREL  (0 1)
          CAR
          CLRCAR
          POSREL  (0 1)
          CAR
```

Now, lets examine the generated object code. Note that the entire code for these macros will be generated each time you expand the CAR or CLRCAR macros. This will take a lot of memory especially if you use CAR or CLRCAR many times.

To create an efficient compiler, lets make as much of the macros as possible a subroutine which can be called. In this manner, we compile a JSR every time a CAR or CLRCAR instruction is written. A good place to put this subroutine part of your macro would be in the BEGIN definition. Since every graphics drawing program must begin with a BEGIN statement, the subroutine code will be generated at the start for your macros to JSR to. Now, lets write the subroutines for placement in BEGIN.

```
@CAR       PRINT (@@CAR)
           RTS
@@CAR      .BY '▄█▄' CD 8 8 8 'O O' 8 8 8 CU 0

@CLRCAR    PRINT (@@CARC)
           RTS
@@CARC     .BY '   ' CD 8 8 8 '   ' 8 8 8 CU 0
```

And their associated Macro definitions (do not put in the BEGIN macro)

```
 !!!CAR      .MD
            JSR @CAR
            .ME

 !!!CLRCAR   .MD
            JSR @CLRCAR
            .ME
```

Observe that only 3 bytes of code (a JSR) will be generated for each use of the instructions CAR and CLRCAR since the BEGIN statement expanded the subroutines.

As a side note, to move the Car 10 positions to the right, you can use a do loop as follows:

```
            DEFINE (J 10)
            DO    (EXIT J)
            CLRCAR
            CAR
            POSREL (0 1)
            END
EXIT
```

You can place your macros in either the macro set that you build a compiler with, or place them in your graphics drawing program. If you place them in your program, they will not be available for use by other programs.

7. GRAPHICS COMPILER INSTRUCTION SET SUMMARY

ADD	(labelI label2)	labelI=labelI+label2
BEGIN		Begin Compile
BELL		Ring bell
CLEAR		Clear to end of screen
DO	(labelI label2)	loop label2 times then go to labelI
DEFINE	(labelI #I)	labelI=#I
DRAWD		Draw line using character
DRAWL	(labelI label2)	at labelI
DRAWR		
DRAWU		
END		Terminal do loop or program
GRAPHN		Graphics = No
GRAPHY		Graphics = Yes
HOME		Home cursor
INPUTB	(labelI)	Input byte and store at labelI
INPUTC	(labelI)	Input ascii char. and store at labelI
JUMP	(labelI)	Jump to labelI
JUMPE		
JUMPG		Jump conditionally
JUMPGE	(labelI label2)	on labelI to
JUMPL		location label2
JUMPLE		
JUMPN		

```
OUTPUTB  (labelI)   Output byte at labelI as 2 hex digits
OUTPUTC  (labelI)   Output ascii character at labelI

POSABS  (labelI label2)   Position cursor at absolute
                          labelI (row), label2 (column)

POSREL  (labelI label2)   Position cursor relatively at
                          labelI (row), label2 (column)

PRINT  (labelI)  Print text at labelI

REVRSN           Reverse video = No
REVRSY           Reverse video = Yes

SETA  (#1)
SETAB (#1 #2)                Store at locations
SETABC (#1 #2 #3)            ↑A,↑B,↑C,↑D
SETABCD (#1 #2 #3 #4)

SUB  (labelI label2)   labelI = labelI - label2

VECTUR
VECTUL      labelI label2 label3 label4)
VECTLR
VECTLL

           Where:  labelI=char. to draw vector
                   label2="rise"
                   label3="run"
                   label4=length
```

8. COMBINING MACHINE LANGUAGE AND BASIC PROGRAMS - PET

BASIC and machine language (ML) programs can be easily combined to function together as one program. They can even be saved and loaded as one program from cassette tape.

The following is a series of guidelines which should be followed when combining BASIC and ML programs. These guidelines assume that both programs have been debugged and saved on tape.

1. After saving the BASIC program, type PRINT PEEK (125)*256+PEEK(124) for old ROMS or PRINT PEEK (43)*256+PEEK(42) for new ROMS. The number printed is the decimal address of the end of the BASIC program. Convert this decimal number into hex since it will be needed when assembling the ML routine.

2. Load the ASSM/TED and the graphics compiler program containing your ML source program. Now, (using the normal .BA and .OS pseudo ops) assemble the ML program so that it will be stored in memory just beyond the last memory location used by BASIC (which was calculated above). After the ML program has been assembled, type >LABEL. Find the starting and ending labels of your program and write down the hex address's for future use. Also, convert the hex address to decimal.

3. Immediately exit the ASSM/TED and monitor and load your BASIC program.
```
Type:     POKE 125, (INT (X/256))
          POKE 124,((X/256)-(INT9X/256)))*256
          for old ROMS
      or
          POKE 43,(INT(X/256))
          POKE 42, ((X/256)-(INT(X/256)))*256
          for new ROMS
```

   Where X is the decimal ending address of the ML program. Now SAVE the program as you normally would.

Note: If you are using a PET with old ROMs, do not assemble and store a program below $0770. The PET monitor in RAM is stored there.

HOW TO TRANSFER BETWEEN BASIC AND YOUR MACHINE LANGUAGE ROUTINE

The easiest way to go to your ML routine from BASIC is via the SYS command (although the USR command may also be used). When using the SYS command in the BASIC portion of the program, care must be taken because no new characters can be added or deleted from any part of the BASIC lines. Thus, when writing the SYS command,type it like SYS(00000). After the programs have been combined, you can LIST the BASIC program and put the address in the SYS command (for example SYS(02897); But remember not to add or delete any character - only change.

9. GRAPHICS COMPILER SOURCE LISTING

   Listing 3A and 3B show the source listings for PET and SYM.

LISTING 1A - Enhancements for Pet versions.

```
1F00   00 00 00 00 00 00 00 00 00 20 02 26 C9 43 F0 61
1F10   48 AD 00 1F D0 56 AD 00 3F 85 3D AD 01 3F 85 3E
1F20   68 C9 4C F0 22 C9 4D F0 03 4C D9 23 20 94 24 A9
1F30   FF 8D 09 3F C0 50 B0 05 A2 08 20 84 22 20 BC 21
1F40   F0 05 B0 03 20 42 23 EE 00 1F A0 07 B9 00 3F 99
1F50   01 1F 88 10 F7 A5 3D 8D 00 3F A5 3E 8D 01 3F A5
1F60   35 8D 04 3F A5 36 8D 05 3F 4C 92 20 A2 30 4C EB
1F70   23 AD 00 1F F0 F6 8E 00 1F A0 07 B9 01 1F 99 00
1F80   3F 88 10 F7 4C 92 20 AD 00 1F F0 0B AD 05 1F 85
1F90   3D AD 06 1F 85 3E 60 4C 5F 24 20 94 24 C0 50 B0
1FA0   13 8E 11 3F A9 01 8D 13 3F 20 81 31 E6 31 A5 31
1FB0   29 1F 85 4A 4C 41 20 A0 00 B9 00 00 99 00 1E C8
1FC0   D0 F7 60 A2 00 BD 00 1E 48 B5 00 8D 00 1E 68 95
1FD0   00 E8 D0 F1 60 20 F2 3E 8E 00 1F 20 B7 1F A9 0B
1FE0   85 4A 60 20 C3 1F 4C 8A 20 20 C3 1F 4C 3F 20 00
.
```

```
2004   20 D5 1F
20B3   4C 9A 1F
2095   4C E9 1F
2374   A6 4A
26AD   E3 1F
2717   42 55 09 1F
3051   20 87 1F
```

LISTING 1B - Enhancements for non-Pet version 1.0

```
4000   00 00 00 00 00 00 00 00 00 20 02 26 C9 43 F0 61
4010   48 AD 00 40 D0 56 AD 00 01 85 DD AD 01 01 85 DE
4020   68 C9 4C F0 22 C9 4D F0 03 4C D9 23 20 94 24 A9
4030   FF 8D 09 01 C0 50 B0 05 A2 08 20 84 22 20 BC 21
4040   F0 05 B0 03 20 42 23 EE 00 40 A0 07 B9 00 01 99
4050   01 40 88 10 F7 A5 DD 8D 00 01 A5 DE 8D 01 01 A5
4060   D5 8D 04 01 A5 D6 8D 05 01 4C 92 20 A2 30 4C EB
4070   23 AD 00 40 F0 F6 8E 00 40 A0 07 B9 01 40 99 00
4080   01 88 10 F7 4C 92 20 AD 00 40 F0 0B AD 05 40 85
4090   DD AD 06 40 85 DE 60 4C 5F 24 8E 00 40 20 F2 3E
40A0   A9 0B 85 EA 60 20 94 24 C0 50 B0 13 8E 11 01 A9
40B0   01 8D 13 01 20 81 31 E6 D1 A5 D1 29 1F 85 EA 4C
40C0   41 20 00
.

2004   20 9A 40
20B3   4C A5 40
2374   A6 EA
2717   42 55 09 40
3051   20 87 40
```

LISTING 1C - Enhancements for non-Pet version 2.0

```
4000   00 00 00 00 00 00 00 00 00 20 90 26 C9 43 F0 61
4010   48 AD 00 40 D0 56 AD 00 01 85 DD AD 01 01 85 DE
4020   68 C9 4C F0 22 C9 4D F0 03 4C 39 24 20 0D 25 A9
4030   FF 8D 09 01 C0 50 B0 05 A2 08 20 E4 22 20 12 22
4040   F0 05 B0 03 20 A2 23 EE 00 40 A0 07 B9 00 01 99
4050   01 40 88 10 F7 A5 DD 8D 00 01 A5 DE 8D 01 01 A5
4060   D5 8D 04 01 A5 D6 8D 05 01 4C 53 20 A2 30 4C 4B
4070   24 AD 00 40 F0 F6 8E 00 40 A0 07 B9 01 40 99 00
4080   01 88 10 F7 4C 53 20 AD 00 40 F0 0B AD 05 40 85
4090   DD AD 06 40 85 DE 60 4C C5 24 8E 00 40 8E 13 01
40A0   A9 0B 85 EA 60 00
>

2018   20 9A 40
27A1   42 55 09 40
3130   20 87 40
```

LISTING 2A - PET PROGRAM EXAMPLE WHICH DRAWS A 3-D BOX

```
5000 ;----- PROGRAM EXAMPLE FOLLOWS ----
5005 ;
5010 ;       DRAW 3 DIMENSIONAL BOX
5015                .BA $800
5020                .OS
5025 ;
5030 J              .DE $33A       ; 2ND CASSETTE BUFFER
5035 K              .DE $33B
5040 LEN            .DE $33C
5045 CHAR           .DE $33D
5050 ONE            .DE $33E
5055 CHARI          .DE $33F
5060 CHAR←          .DE $340
5065 CHAR/          .DE $341
5070 LEN1           .DE $342
5075 N              .DE $343
5080 T3             .DE $344
5085 TI             .DE $345
5090 BYTE           .DE $346
5095 ;
5100 BOX            BEGIN
5105                REVRSN
5110                GRAPHY
5115                DEFINE (CHARI $2A)
5120                DEFINE (CHAR/ $2A)
5125                DEFINE (CHAR← $2A)
5130                DEFINE (T3 2)
5135                DEFINE (TI 12)
5140                DO (EXIT T3)
5145                CLEAR
5150                PRINT (MESS1)
5155                INPUTB (LEN)
5160                PRINT (MESS2)
5165                INPUTB (LEN1)
5170                SETAB (10 6)
5175                POSABS (↑A ↑B)
5180                SETAB (1 1)
5185                VECTUR (CHAR/ ↑A ↑B LEN)
5190                DRAWR (CHAR← LEN1)
5195                VECTLL (CHAR/ ↑A ↑B LEN)
5200                REVRSN
5205                DRAWL (CHAR← LEN1)
5210                DRAWD (CHARI LEN)
5215                DRAWR (CHAR← LEN1)
5220                DRAWU (CHARI LEN)
5225                DRAWD (CHARI LEN)
5230                VECTUR (CHAR/ ↑A ↑B LEN)
5235                DRAWU (CHARI LEN)
5240                WAIT (TI)
5245                BELL
5250                END
5255 EXIT           SETAB (22 0)
5260                POSABS (↑A ↑B)
5265                END
0B15- 49 4E 50  5270 MESS1          .BY 'INPUT HEIGHT? ' 0
```

LISTING 2A (cond.) - PET PROGRAM EXAMPLE WHICH DRAWS A 3-D BOX

```
0B18-  55 54 20
0B1B-  48 45 49
0B1E-  47 48 54
0B21-  3F 20 00
0B24-  0D 49 4E   5275 MESS2       .BY $0D 'INPUT WIDTH? ' 0
0B27-  50 55 54
0B2A-  20 57 49
0B2D-  44 54 48
0B30-  3F 20 00
                  5280 ;
                  5285             .EN


LABEL FILE:  [ / = EXTERNAL ]


/J=033A              /K=033B              /LEN=033C
/CHAR=033D           /ONE=033E            /CHARI=033F
/CHAR←=0340          /CHAR/=0341          /LEN1=0342
/N=0343              /T3=0344             /TI=0345
/BYTE=0346           BOX=0800             ↑CHAR=0803
↑LEN=0804            ↑H=0805              ↑V=0806
↑A=0807              ↑B=0808              ↑C=0809
↑D=080A              ↑E=080B              ↑F=080C
↑RVS=080D            /↑WRT.=FFD2          /↑C/L=0028
/↑L/S=0018           /↑LINE=00D8          /↑COL=00C6
/↑GETCHR=FFE4        /↑CLOCK0=008F        ↑HOME=080E
↑CLEAR=0814          ↑FORMROW=081A        @LPCK1=081D
↑FORMCOL=0829        @LPCK2=082E          ↑POSABS=083A
↑POSREL=084F         ↑RVSTEST=0866        ↑GRAPHY=086F
↑GRAPHN=0875         ↑REVRSY=087B         ↑SETRVS=0880
↑REVRSN=0886         ↑DRAWR=0891          ↑DRAWL=089C
↑DRAWD=08A8          ↑DRAWU=08B3          ↑VECTUR=08BE
↑VECTUL=08DA         ↑VECTLL=08F9         ↑VECTLR=0915
↑PRMD=092E           ↑BEEP=0935           ↑SCROLL=095C
↑INPUTB=0974         ↑WAIT=0992           ↑OUTPUTB=09A2
↑INPUTC=09BA         EXIT=0B01            MESS1=0B15
MESS2=0B24
//0000,0B33,0B33
>
```

LISTING 2B - SYM Program Example which draws a 3-D Box.

```
>ASSEMBLE  LIST

                  5000 ;DRAW 3 DIMENSIONAL BOX
                  5010 J           .DE $190
                  5020 K           .DE $191
                  5030 LEN         .DE $192
                  5040 CHAR        .DE $193
                  5050 ONE         .DE $194
                  5060 CHARI       .DE $195
                  5070 CHAR←       .DE $196
                  5080 CHAR/       .DE $197
                  5090 LEN1        .DE $198
                  5100
                  5110             .BA $300
                  5120             .OS
                  5140 BOX         BEGIN
                  5150             HOME
                  5160             CLEAR
                  5170             BELL
                  5180             PRINT (MESS1)
                  5190             INPUTB (LEN)
                  5200             BELL
                  5210             PRINT (MESS2)
                  5220             INPUTB (LEN1)
                  5230             BELL
                  5240 X           REVRSY
                  5250             GRAPHY
                  5260             DEFINE (CHARI $55)
                  5270             DEFINE (CHAR/ $6F)
                  5280             DEFINE (CHAR← $4F)
                  5290
                  5300             SETAB (12 10)
                  5310             POSABS (↑A ↑B)
                  5320             SETAB (1 1)
                  5330             VECTUR (CHAR/ ↑A ↑B LEN)
                  5340             DRAWR (CHAR← LEN1)
                  5350             VECTLL (CHAR/ ↑A ↑B LEN)
                  5360             DRAWL (CHAR← LEN1)
                  5370             DRAWD (CHARI LEN)
                  5380             DRAWR (CHAR← LEN1)
                  5390             DRAWU (CHARI LEN)
                  5400             DRAWD (CHARI LEN)
                  5410             VECTUR (CHAR/ ↑A ↑B LEN)
                  5420             DRAWU (CHARI LEN)
                  5430
                  5440 EXIT        GRAPHN
                  5450             REVRSN
                  5460             SETAB (21 1)
                  5470             POSABS (↑A ↑B)
                  5480             END
                  5490
0568- 49 4E 50    5500 MESS1       .BY 'INPUT HEIGHT THEN WIDTH? ' 0
056B- 55 54 20
056E- 48 45 49
0571- 47 48 54
0574- 20 54 48
0577- 45 4E 20
057A- 57 49 44
```

LISTING 2B (cond.) - SYM Program Example which draws a 3-D Box.

```
057D-  54 48 3F
0580-  20 00
0582-  20 00        5510 MESS2       .BY ' ' 0
                    5520             .EN

LABEL FILE:  [ / = EXTERNAL ]

/J=0190                    /K=0191                    /LEN=0192
/CHAR=0193                 /ONE=0194                  /CHARI=0195
/CHAR+=0196                /CHAR/=0197                /LEN1=0198
BOX=0300                   ↑CHAR=030B                 ↑LEN=030C
↑H=030D                    ↑V=030E                    ↑A=030F
↑B=0310                    ↑C=0311                    ↑D=0312
/↑WRT.=8A47                /↑ESC=001B                 /↑C/L=0050
/↑L/S=0018                 /↑BEEP=8972                ↑HOME=0313
↑CLEAR=031E                ↑POSREL=0329               ↑POSABS=0343
↑GRAPHY=035D               ↑GRAPHN=0368               ↑REVRSY=0373
↑REVRSN=037E               ↑DRAWR=0389                ↑DRAWL=0394
↑DRAWD=03A0                ↑DRAWU=03AB                ↑VECTUR=03B6
↑VECTUL=03D2               ↑VECTLL=03F1               ↑VECTLR=040D
↑PRMD=0426                 X=0462                     EXIT=054E
MESS1=0568                 MESS2=0582
//0000,0584,0584
>
```

LISTING 3A - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
;◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
;◆◆◆ GRAPHICS COMPILER FOR PET ◆◆◆◆
;◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆
;
;          COPYRIGHT 1979
;      C.W. MOSER & J.R. HALL
;
;  VERSION 10/1
;
;
;
!!!@HOME    .MD
↑HOME       LDA #$13
            JSR ↑WRT.
            RTS
            .ME
;
!!!@CLEAR   .MD
↑CLEAR      LDA #$93
            JSR ↑WRT.
            RTS
            .ME
;
;A=ROW   Y=COL
!!!@POSABS  .MD
↑POSABS     JSR @LPCK1
            LDA #$0D
            JSR ↑WRT.
            LDA #$91
            JSR ↑WRT.
            TYA
            JSR @LPCK2
            JSR ↑RVSTEST
            RTS
            .ME
;
;A=ROW   Y=COL
!!!@POSREL  .MD
↑POSREL     PHA
            LDA ◆↑COL
            STA ↑E
            PLA
            JSR ↑FORMROW
            LDA #$0D
            JSR ↑WRT.
            LDA #$91
            JSR ↑WRT.
            JSR ↑FORMCOL
↑RVSTEST    LDA ↑RVS
            BEQ =+4
            JSR ↑SETRVS
            RTS
            .ME
;
!!!@FRMROW  .MD
```

LISTING 3A(cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
↑FORMROW     CLC
             ADC ◆↑LINE
@LPCK1       CMP #$18
             BCC ...SKIP1
             SBC #$18
             JMP @LPCK1
...SKIP1     STA ◆↑LINE
             RTS
             .ME
;
!!!@FRMCOL   .MD
↑FORMCOL     TYA
             CLC
             ADC ↑E
@LPCK2       CMP #$28
             BCC ...SKIP2
             SBC #$28
             JMP @LPCK2
...SKIP2     STA ◆↑COL
             RTS
             .ME
;
!!!@GRAPHY   .MD
↑GRAPHY      LDA #$C
             STA $E84C
             RTS
             .ME
;
!!!@GRAPHN   .MD
↑GRAPHN      LDA #$E
             STA $E84C
             RTS
             .ME
;
!!!@REVRSY   .MD
↑REVRSY      LDA #$1
             STA ↑RVS
↑SETRVS      LDA #$12
             JSR ↑WRT.
             RTS
             .ME
;
!!!@REVRSN   .MD
↑REVRSN      LDA #0
             STA ↑RVS
             LDA #$92
             JSR ↑WRT.
             RTS
             .ME
;
!!!@PRMD     .MD
↑PRMD        STA ↑V
             STY ↑H
             RTS
             .ME
;
!!!@DRAWR    .MD
↑DRAWR       LDA #00
```

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
0570                LDY #01
0575                JSR ↑PRMD
0580                JSR ↑VECTLR
0585                RTS
0590                .ME
0595 ;
0600 !!!@DRAWL      .MD
0605 ↑DRAWL         LDY #↑L/S
0610                TYA
0615                LDY #01
0620                JSR ↑PRMD
0625                JSR ↑VECTUL
0630                RTS
0635                .ME
0640 ;
0645 !!!@DRAWD      .MD
0650 ↑DRAWD         LDA #01
0655                LDY #↑C/L
0660                JSR ↑PRMD
0665                JSR ↑VECTLR
0670                RTS
0675                .ME
0680 ;
0685 !!!@DRAWU      .MD
0690 ↑DRAWU         LDA #01
0695                LDY #↑C/L
0700                JSR ↑PRMD
0705                JSR ↑VECTUR
0710                RTS
0715                .ME
0720 ;
0725 !!!@VECTUR     .MD
0730 ↑VECTUR        LDX ↑LEN
0735                BEQ ...EXVUR
0740 ...LPVUR       LDA ↑CHAR
0745                JSR ↑WRT.
0750                LDY ↑H
0755                DEY
0760                LDA #↑L/S
0765                SEC
0770                SBC ↑V
0775                JSR ↑POSREL
0780                DEX
0785                BNE ...LPVUR
0790 ...EXVUR       RTS
0795                .ME
0800 ;
0805 !!!@VECTUL     .MD
0810 ↑VECTUL        LDX ↑LEN
0815                BEQ ...EXVUL
0820 ...LPVUL       LDA ↑CHAR
0825                JSR ↑WRT.
0830                LDA #↑C/L
0835                CLC
0840                SBC ↑H
0845                TAY
0850                LDA #↑L/S
0855                SEC
```

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
0860                SBC ↑V
0865                JSR ↑POSREL
0870                DEX
0875                BNE ...LPVUL
0880 ...EXVUL       RTS
0885                .ME
0890 ;
0895 !!!@VECTLL     .MD
0900 ↑VECTLL        LDX ↑LEN
0905                BEQ ...EXVLL
0910 ...LPVLL       LDA ↑CHAR
0915                JSR ↑WRT.
0920                LDA #↑C/L
0925                CLC
0930                SBC ↑H
0935                TAY
0940                LDA ↑V
0945                JSR ↑POSREL
0950                DEX
0955                BNE ...LPVLL
0960 ...EXVLL       RTS
0965                .ME
0970 ;
0975 !!!@VECTLR     .MD
0980 ↑VECTLR        LDX ↑LEN
0985                BEQ ...EXVUL
0990 ...LPVLR       LDA ↑CHAR
0995                JSR ↑WRT.
1000                LDY ↑H
1005                DEY
1010                LDA ↑V
1015                JSR ↑POSREL
1020                DEX
1025                BNE ...LPVLR
1030 ...EXVLR       RTS
1035 ;
1040                .ME
1045 ;
1050 !!!@BEEP       .MD
1055 ↑BEEP          LDA #$10
1060                STA $E84B
1065                LDA #$33
1070                STA $E84A
1075                LDA #$FB
1080                STA $E848
1085                LDY #$55
1090 ...DE1         LDX #$55
1095 ...DELAY       PHA
1100                PLA
1105                DEX
1110                BNE ...DELAY
1115                DEY
1120                BNE ...DE1
1125                LDA #$0
1130                STA $E84B
1135                STA $E84A
1140                STA $E848
1145                RTS
```

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
                .ME
;
!!!@INPUTB      .MD
↑INPUTB         JSR ...NIBBLE
                CLC
                ROL A
                ROL A
                ROL A
                ROL A
                STA ↑F
                JSR ...NIBBLE
                ORA ↑F
                RTS
...NIBBLE       JSR ↑INPUTC
                CMP #$3A
                BCC ...SKIP
                ADC #$08
...SKIP         AND #$0F
                RTS
                .ME
;
!!!@OUTPTB      .MD
↑OUTPUTB        PHA
                ROR A
                ROR A
                ROR A
                ROR A
                JSR ...NIB
                PLA
...NIB          AND #$0F
                CMP #$0A
                BCC ...PASS
                ADC #$06
...PASS         CLC
                ADC #$30
                JSR ↑WRT.
                RTS
                .ME
;
!!!@INPUTC      .MD
↑INPUTC         JSR ↑GETCHR
                CMP #0
                BEQ ↑INPUTC
                JSR ↑WRT.
                RTS
                .ME
;
!!!@WAIT        .MD
↑WAIT           LDA #0
                STA ↑CLOCK0
                LDA #15
...WALOP        CMP ↑CLOCK0
                BNE ...WALOP
                DEX
                BNE ↑WAIT
                RTS
                .ME
;
```

PAGE 06

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
1440 !!!DO        .MD (...EXDO ...L)
1445              LDA ...L
1450              BEQ ...EXDO1
1455 ...LPDO      JSR ...DOLOOP
1460              DEC ...L
1465              BNE ...LPDO
1470 ...EXDO1     JMP ...EXDO
1475 ...DOLOOP    .ME
1480
1485 !!!END       .MD
1490              RTS
1495              .ME
1500 ;
1505 !!!SUB       .MD (...LABD ...D)
1510              LDA ...LABD
1515              SEC
1520              SBC ...D
1525              STA ...LABD
1530              .ME
1535 ;
1540 !!!ADD       .MD (...LABU ...U)
1545              LDA ...LABU
1550              CLC
1555              ADC ...U
1560              STA ...LABU
1565              .ME
1570 ;
1575 !!!DEFINE    .MD (...LDEF ...V)
1580              LDA #...V
1585              STA ...LDEF
1590              .ME
1595 ;
1600 !!!JUMPE     .MD (...LTEST ...LJMPE)
1605              LDA ...LTEST
1610              BNE ...SKJE
1615              JMP ...LJMPE
1620 ...SKJE      .ME
1625 ;
1630 !!!JUMPN     .MD (...LTEST ...LJMPN)
1635              LDA ...LTEST
1640              BEQ ...SKJN
1645              JMP ...LJMPN
1650 ...SKJN      .ME
1655 ;
1660 !!!JUMPL     .MD (...LTEST ...LJMPL)
1665              LDA ...LTEST
1670              BPL ...SKJL
1675              JMP ...LJMPL
1680 ...SKJL      .ME
1685 ;
1690 !!!JUMPG     .MD (...LTEST ...LJMPG)
1695              LDA ...LTEST
1700              BMI ...SKJG
1705              BEQ ...SKJG
1710              JMP ...LJMPG
1715 ...SKJG      .ME
1720 ;
1725 !!!JUMPGE    .MD (...LTEST ...LJMPGE)
```

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
                LDA ...LTEST
                BMI ...SKJGE
                JMP ...LJMPGE
...SKJGE        .ME
;
!!!JUMPLE       .MD (...LTEST ...LJMPLE)
                LDA ...LTEST
                BEQ ...SKJLE1
                BPL ...SKJLE2
...SKJLE1       JMP ...LJMPLE
...SKJLE2       .ME
;
!!!@DPRM        .MD (...C ...L)
                LDA ...C
                STA ↑CHAR
                LDA ...L
                STA ↑LEN
                .ME
;
!!!@VPRM        .MD (...C ...V ...H ...L)
                LDA ...C
                STA ↑CHAR
                LDA ...V
                STA ↑V
                LDA ...H
                STA ↑H
                LDA ...L
                STA ↑LEN
                .ME
;
!!!@SCROLL      .MD
↑SCROLL         LDA #$17
                LDY #$0
                JSR ↑POSABS
                LDA ↑A
                STA ↑H
                LDA #$11
...AGAIN        JSR ↑WRT.
                DEC ↑H
                BNE ...AGAIN
                RTS
                .ME
;
!!!HOME         .MD
                JSR ↑HOME
                .ME
;
!!!CLEAR        .MD
                JSR ↑CLEAR
                .ME
;
!!!POSREL       .MD (...J ...K)
                LDA ...J
                LDY ...K
                JSR ↑POSREL
                .ME
;
!!!POSABS       .MD ( ...X ...Y )
```

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
2020              LDA ...X
2025              LDY ...Y
2030              JSR ↑POSABS
2035              .ME
2040 ;
2045 !!!GRAPHY    .MD
2050              JSR ↑GRAPHY
2055              .ME
2060 ;
2065 !!!GRAPHN    .MD
2070              JSR ↑GRAPHN
2075              .ME
2080 ;
2085 !!!REVRSY    .MD
2090              JSR ↑REVRSY
2095              .ME
2100 ;
2105 !!!REVRSN    .MD
2110              JSR ↑REVRSN
2115              .ME
2120 ;
2125 !!!BELL      .MD
2130              JSR ↑BEEP
2135              .ME
2140 ;
2145 !!!DRAWR     .MD (...C ...L)
2150             @DPRM (...C ...L)
2155              JSR ↑DRAWR
2160              .ME
2165 ;
2170 !!!DRAWL     .MD (...C ...L)
2175             @DPRM (...C ...L)
2180              JSR ↑DRAWL
2185              .ME
2190 ;
2195 !!!DRAWD     .MD (...C ...L)
2200             @DPRM (...C ...L)
2205              JSR ↑DRAWD
2210              .ME
2215 ;
2220 !!!DRAWU     .MD (...C ...L)
2225             @DPRM (...C ...L)
2230              JSR ↑DRAWU
2235              .ME
2240 ;
2245 !!!VECTUR    .MD (...C ...V ...H ...L)
2250             @VPRM (...C ...V ...H ...L)
2255              JSR ↑VECTUR
2260              .ME
2265 ;
2270 !!!VECTUL    .MD (...C ...V ...H ...L)
2275             @VPRM (...C ...V ...H ...L)
2280              JSR ↑VECTUL
2285              .ME
2290 ;
2295 !!!VECTLL    .MD (...C ...V ...H ...L)
2300             @VPRM (...C ...V ...H ...L)
2305              JSR ↑VECTLL
```

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
           .ME
;
!!!VECTLR  .MD (...C ...V ...H ...L)
           @VPRM (...C ...V ...H ...L)
           JSR ↑VECTLR
           .ME
;
!!!SCROLL  .MD
           JSR ↑SCROLL
           .ME
;
!!!INPUTB  .MD (...R)
           JSR ↑INPUTB
           STA ...R
           .ME
;
!!!WAIT    .MD (...W)
           LDX ...W
           JSR ↑WAIT
           .ME
;
!!!OUTPUTB .MD (...B)
           LDA ...B
           JSR ↑OUTPUTB
           .ME
;
!!!INPUTC  .MD (...C)
           JSR ↑INPUTC
           STA ...C
           .ME
;
!!!OUTPUTC .MD (...C)
           LDA ...C
           JSR ↑WRT.
           .ME
;
!!!BEGIN   .MD
           JMP ...BEG
↑CHAR      .DS 1
↑LEN       .DS 1
↑H         .DS 1
↑V         .DS 1
↑A         .DS 1
↑B         .DS 1
↑C         .DS 1
↑D         .DS 1
↑E         .DS 1
↑F         .DS 1
↑RVS       .DS 1
↑WRT.      .DE $FFD2
↑C/L       .DE 40
↑L/S       .DE 24
↑LINE      .DE $D8         ;IF OLD ROMS, CHANGE D8 TO F5

↑COL       .DE $C6         ;IF OLD ROMS, CHANGE C6 TO E2

↑GETCHR    .DE $FFE4
↑CLOCK0    .DE $8F         ;IF OLD ROMS, CHANGE 8F TO 202

;
           @HOME
```

LISTING 3A (cond.) - SOURCE MACROS FOR PET GRAPHICS DRAWING COMPILER

```
                   @CLEAR
                   @FRMROW
                   @FRMCOL
                   @POSABS
                   @POSREL
                   @GRAPHY
                   @GRAPHN
                   @REVRSY
                   @REVRSN
                   @DRAWR
                   @DRAWL
                   @DRAWD
                   @DRAWU
                   @VECTUR
                   @VECTUL
                   @VECTLL
                   @VECTLR
                   @PRMD
                   @BEEP
                   @SCROLL
                   @INPUTB
                   @WAIT
                   @OUTPTB
                   @INPUTC
;
...BEG             .ME
;
!!!SETA            .MD (...A)
                   LDA #...A
                   STA ↑A
                   .ME
;
!!!SETAB           .MD (...A ...B)
                   LDA #...A
                   STA ↑A
                   LDA #...B
                   STA ↑B
                   .ME
;
!!!SETABC          .MD (...A ...B ...C)
                   SETAB (...A ...B)
                   LDA #...C
                   STA ↑C
                   .ME
;
!!!SETABCD         .MD (...A ...B ...C ...D)
                   SETABC (...A ...B ...C)
                   LDA #...D
                   STA ↑D
                   .ME
;
!!!PRINT           .MD (...M)
                   LDY #0
...LPPR            LDA ...M,Y
                   BEQ ...EXPR
                   JSR ↑WRT.
                   INY
                   BNE ...LPPR
...EXPR            .ME
;
                   .EN
```

↑X

LISTING 3B - Source Macros for Graphics Drawing Compiler.

```
>ASSEMBLE  LIST

                0000 ;*** GRAPHICS DRAWING MACROS FOR SYM-1 WITH KTM 2/80 ***
                0001 ;
                0010 !!!@HOME    .MD
                0020 ↑HOME       LDA #↑ESC
                0030             JSR ↑WRT.
                0040             LDA #'H
                0050             JSR ↑WRT.
                0060             RTS
                0070             .ME
                0080
                0090 !!!@CLEAR   .MD
                0100 ↑CLEAR      LDA #↑ESC
                0110             JSR ↑WRT.
                0120             LDA #'J
                0130             JSR ↑WRT.
                0140             RTS
                0150             .ME
                0160
                0170 ;A=ROW    Y=COL
                0180 !!!@POSREL  .MD
                0190 ↑POSREL     CLC
                0200             ADC #'
                0210             PHA
                0220             LDA #↑ESC
                0230             JSR ↑WRT.
                0240             LDA #'+
                0250             JSR ↑WRT.
                0260             PLA
                0270             JSR ↑WRT.
                0280             TYA
                0290             CLC
                0300             ADC #'          ;ADJUST COLUMN
                0310             JSR ↑WRT.
                0320             RTS
                0330             .ME
                0340
                0350 ;A=ROW    Y=COL
                0360 !!!@POSABS  .MD
                0370 ↑POSABS     CLC
                0380             ADC #'
                0390             PHA
                0400             LDA #↑ESC
                0410             JSR ↑WRT.
                0420             LDA #'=
                0430             JSR ↑WRT.
                0440             PLA
                0450             JSR ↑WRT.
                0460             TYA
                0470             CLC
                0480             ADC #'
                0490             JSR ↑WRT.
                0500             RTS
                0510             .ME
                0520
                0530 !!!@GRAPHY  .MD
                0540 ↑GRAPHY     LDA #↑ESC
```

LISTING 3B (cond.) - Source Macros for Graphics Drawing Compiler.

```
0550                JSR ↑WRT.
0560                LDA #'G
0570                JSR ↑WRT.
0580                RTS
0590                .ME
0600
0610 !!!@GRAPHN     .MD
0620 ↑GRAPHN        LDA #↑ESC
0630                JSR ↑WRT.
0640                LDA #'G
0650                JSR ↑WRT.
0660                RTS
0670                .ME
0680
0690 !!!@REVRSY     .MD
0700 ↑REVRSY        LDA #↑ESC
0710                JSR ↑WRT.
0720                LDA #'R
0730                JSR ↑WRT.
0740                RTS
0750                .ME
0760
0770 !!!@REVRSN     .MD
0780 ↑REVRSN        LDA #↑ESC
0790                JSR ↑WRT.
0800                LDA #'R
0810                JSR ↑WRT.
0820                RTS
0830                .ME
0840
0850 !!!@PRMD       .MD
0860 ↑PRMD          STA ↑V
0870                STY ↑H
0880                RTS
0890                .ME
0900
0910 !!!@DRAWR      .MD
0920 ↑DRAWR         LDA #00
0930                LDY #01
0940                JSR ↑PRMD
0950                JSR ↑VECTLR
0960                RTS
0970                .ME
0980
0990 !!!@DRAWL      .MD
1000 ↑DRAWL         LDY #↑L/S
1010                TYA
1020                LDY #01
1030                JSR ↑PRMD
1040                JSR ↑VECTUL
1050                RTS
1060                .ME
1070
1080 !!!@DRAWD      .MD
1090 ↑DRAWD         LDA #01
1100                LDY #↑C/L
1110                JSR ↑PRMD
1120                JSR ↑VECTLR
```

LISTING 3B (cond.) - Source Macros for Graphics Drawing Compiler.

```
                      RTS
                      .ME

          !!!@DRAWU   .MD
          ↑DRAWU      LDA #01
                      LDY #↑C/L
                      JSR ↑PRMD
                      JSR ↑VECTUR
                      RTS
                      .ME

          !!!@VECTUR  .MD
          ↑VECTUR     LDX ↑LEN
                      BEQ ...EXVUR
          ...LPVUR    LDA ↑CHAR
                      JSR ↑WRT.
                      LDY ↑H
                      DEY
                      LDA #↑L/S
                      SEC
                      SBC ↑V
                      JSR ↑POSREL
                      DEX
                      BNE ...LPVUR
          ...EXVUR    RTS
                      .ME

          !!!@VECTUL  .MD
          ↑VECTUL     LDX ↑LEN
                      BEQ ...EXVUL
          ...LPVUL    LDA ↑CHAR
                      JSR ↑WRT.
                      LDA #↑C/L
                      CLC
                      SBC ↑H
                      TAY
                      LDA #↑L/S
                      SEC
                      SBC ↑V
                      JSR ↑POSREL
                      DEX
                      BNE ...LPVUL
          ...EXVUL    RTS
                      .ME

          !!!@VECTLL  .MD
          ↑VECTLL     LDX ↑LEN
                      BEQ ...EXVLL
          ...LPVLL    LDA ↑CHAR
                      JSR ↑WRT.
                      LDA #↑C/L
                      CLC
                      SBC ↑H
                      TAY
                      LDA ↑V
                      JSR ↑POSREL
                      DEX
                      BNE ...LPVLL
```

LISTING 3B (cond.) - Source Macros for Graphics Drawing Compiler.

```
...EXVLL    RTS
            .ME

!!!@VECTLR  .MD
↑VECTLR     LDX ↑LEN
            BEQ ...EXVUL
...LPVLR    LDA ↑CHAR
            JSR ↑WRT.
            LDY ↑H
            DEY
            LDA ↑V
            JSR ↑POSREL
            DEX
            BNE ...LPVLR
...EXVLR    RTS
            .ME

!!!DO       .MD (...EXDO ...L)
            LDA ...L
            BEQ ...EXDO1
...LPDO     JSR ...DOLOOP
            DEC ...L
            BNE ...LPDO
...EXDO1    JMP ...EXDO
...DOLOOP   .ME

!!!END      .MD
            RTS
            .ME

!!!SUB      .MD (...LABD ...D)
            LDA ...LABD
            SEC
            SBC ...D
            STA ...LABD
            .ME

!!!ADD      .MD (...LABU ...U)
            LDA ...LABU
            CLC
            ADC ...U
            STA ...LABU
            .ME

!!!DEFINE   .MD (...LDEF ...V)
            LDA #...V
            STA ...LDEF
            .ME

!!!JUMP     .MD (...LJMP)
            JMP ...LJMP
            .ME

!!!JUMPE    .MD (...LTEST ...LJMPE)
            LDA ...LTEST
            BNE ...SKJE
            JMP ...LJMPE
...SKJE     .ME
```

LISTING 3B (cond.) - Source Macros for Graphics Drawing Compiler

```
2290
2300 !!!JUMPN      .MD (...LTEST ...LJMPN)
2310               LDA ...LTEST
2320               BEQ ...SKJN
2330               JMP ...LJMPN
2340 ...SKJN       .ME
2350
2360 !!!JUMPL      .MD (...LTEST ...LJMPL)
2370               LDA ...LTEST
2380               BPL ...SKJL
2390               JMP ...LJMPL
2400 ...SKJL       .ME
2410
2420 !!!JUMPG      .MD (...LTEST ...LJMPG)
2430               LDA ...LTEST
2440               BMI ...SKJG
2450               BEQ ...SKJG
2460               JMP ...LJMPG
2470 ...SKJG       .ME
2480
2490 !!!JUMPGE     .MD (...LTEST ...LJMPGE)
2500               LDA ...LTEST
2510               BMI ...SKJGE
2520               JMP ...LJMPGE
2530 ...SKJGE      .ME
2540
2550 !!!JUMPLE     .MD (...LTEST ...LJMPLE)
2560               LDA ...LTEST
2570               BEQ ...SKJLE1
2580               BPL ...SKJLE2
2590 ...SKJLE1     JMP ...LJMPLE
2600 ...SKJLE2     .ME
2610
2620 !!!@DPRM      .MD (...C ...L)
2630               LDA ...C
2640               STA ↑CHAR
2650               LDA ...L
2660               STA ↑LEN
2670               .ME
2680
2690 !!!@VPRM      .MD (...C ...V ...H ...L)
2700               LDA ...C
2710               STA ↑CHAR
2720               LDA ...V
2730               STA ↑V
2740               LDA ...H
2750               STA ↑H
2760               LDA ...L
2770               STA ↑LEN
2780               .ME
2790
2800 !!!HOME       .MD
2810               JSR ↑HOME
2820               .ME
2830
2840 !!!CLEAR      .MD
2850               JSR ↑CLEAR
2860               .ME
```

LISTING 3B (cond.) - Source Macros for Graphics Drawing Compiler.

```

!!!POSREL  .MD (...J ...K)
           LDA ...J
           LDY ...K
           JSR ↑POSREL
           .ME

!!!POSABS  .MD ( ...X ...Y )
           LDA ...X
           LDY ...Y
           JSR ↑POSABS
           .ME

!!!GRAPHY  .MD
           JSR ↑GRAPHY
           .ME

!!!GRAPHN  .MD
           JSR ↑GRAPHN
           .ME

!!!REVRSY  .MD
           JSR ↑REVRSY
           .ME

!!!REVRSN  .MD
           JSR ↑REVRSN
           .ME

!!!BELL    .MD
           JSR ↑BEEP
           .ME

!!!DRAWR   .MD (...C ...L)
           ƏDPRM (...C ...L)
           JSR ↑DRAWR
           .ME

!!!DRAWL   .MD (...C ...L)
           ƏDPRM (...C ...L)
           JSR ↑DRAWL
           .ME

!!!DRAWD   .MD (...C ...L)
           ƏDPRM (...C ...L)
           JSR ↑DRAWD
           .ME

!!!DRAWU   .MD (...C ...L)
           ƏDPRM (...C ...L)
           JSR ↑DRAWU
           .ME

!!!VECTUR  .MD (...C ...V ...H ...L)
           ƏVPRM (...C ...V ...H ...L)
           JSR ↑VECTUR
           .ME

```

LISTING 3B (cond.) - Source Macros for Graphics Drawing Compiler.

```
3450 !!!VECTUL   .MD (...C ...V ...H ...L)
3460             @VPRM (...C ...V ...H ...L)
3470             JSR ↑VECTUL
3480             .ME
3490
3500 !!!VECTLL   .MD (...C ...V ...H ...L)
3510             @VPRM (...C ...V ...H ...L)
3520             JSR ↑VECTLL
3530             .ME
3540
3550 !!!VECTLR   .MD (...C ...V ...H ...L)
3560             @VPRM (...C ...V ...H ...L)
3570             JSR ↑VECTLR
3580             .ME
3590
3600 !!!BEGIN    .MD
3610             JSR $8B86
3620             LDA #$80
3630             STA $A653
3640
3650             JMP ...BEG
3660 ↑CHAR       .DS 1
3670 ↑LEN        .DS 1
3680 ↑H          .DS 1
3690 ↑V          .DS 1
3700 ↑A          .DS 1
3710 ↑B          .DS 1
3720 ↑C          .DS 1
3730 ↑D          .DS 1
3740 ↑WRT.       .DE $8A47
3750 ↑ESC        .DE $1B
3760 ↑C/L        .DE 80
3770 ↑L/S        .DE 24
3780 ↑BEEP       .DE $8972
3790
3800             @HOME
3810             @CLEAR
3820             @POSREL
3830             @POSABS
3840             @GRAPHY
3850             @GRAPHN
3860             @REVRSY
3870             @REVRSN
3880             @DRAWR
3890             @DRAWL
3900             @DRAWD
3910             @DRAWU
3920             @VECTUP
3930             @VECTUL
3940             @VECTLL
3950             @VECTLR
3960             @PRMD
3970
3980 ...BEG
3990
4000             .ME
4010
4020 !!!SETA     .MD (...A)
```

LISTING 3B (cond.) - Source Macros for Graphics Drawing Compiler.

```
                  LDA #...A
                  STA ↑A
                  .ME

!!!SETAB          .MD (...A ...B)
                  LDA #...A
                  STA ↑A
                  LDA #...B
                  STA ↑B
                  .ME

!!!SETABC         .MD (...A ...B ...C)
                  SETAB (...A ...B)
                  LDA #...C
                  STA ↑C
                  .ME

!!!SETABCD        .MD (...A ...B ...C ...D)
                  SETABC (...A ...B ...C)
                  LDA #...D
                  STA ↑D
                  .ME

!!!PRINT          .MD (...M)
                  LDY #0
...LPPR           LDA ...M,Y
                  BEQ ...EXPR
                  JSR ↑WRT.
                  INY
                  BNE ...LPPR
...EXPR           .ME


!!!OUTPUTC        .MD (...R1)
                  LDA ...R1
                  JSR $A663
                  .ME


!!!OUTPUTB        .MD (...R2)
                  LDA ...R2
                  JSR $82FA
                  .ME


!!!INPUTC         .MD (...R3)
                  JSR $A660
                  STA ...R3
                  .ME


!!!INPUTB         .MD (...R4)
                  JSR $81D9
                  STA ...R4
                  .ME


                  .EN
```